Extended CS Bridge - Summer 2018 > Tests & Quizzes

Tests & Quizzes

Exam 2

Return to Assessment List

Part 1 of 6 - 15.0 Points

Analyze its running time of **func1** and **func2**.

Give your answering terms of asymptotic order. That is $T(n) = \theta(n^2)$, or $T(n) = \theta(\sqrt{n})$, etc.

Explain your answers.

Question 1 of 2 8.0 Points

```
int* func1(int n){
    int i, j;
    int* arr;

    arr = new int[n];

    for (i = 0; i < n; i++)
        arr[i] = 0;

    for (i = 0; i < n; i++)
        for (j = i; j < n; j++)
            arr[j]++;

    return arr;
}
```

func1 has running time of theta(n^2)

Since for n times in the first loop, the second loop has to execute:

When i=0, it traverses n times
When i=1, it traverses (n-1) times
So on, so forth.

The total number of operations performed is 1+2+3+....n, which is n*(n+1)/2, meaning the asymptotic order is n^2.

Question 2 of 2

7.0 Points

```
int func2(int* arr, int n){
    int i, j;

    for (i = 1; i <= n; i *= 2)
        reverseArray(arr, i);
}

void reverseArray(int* arr, int n){
    int left, right, temp;

    for (left = 0, right = n-1; left <= right; left++, right--){
        temp = arr[left];
        arr[left] = arr[right];
        arr[left] = temp;
    }
}
```

func2 has a running time of theta(n^2).

The reverseArray function has a running time of theta(n), even though the number of operations it performs is only n/2 in the case n is even and (n-1)/2 in the case n is odd, as it's swapping the leftmost and rightmost numbers.

func2 traverses reverseArray with i incrementing exponentially. When i goes from 1 to 2 to 4 to 8..., the order of growth is theta(n).

Since we're traversing reverseArray with an order of growth of n, and reverseArray itself has an order growth of n, func2 has an order growth of (n^2).

Comment: the first for loop doesn't get run n times. This is theta(n).

Part 2 of 6 -

10.0 Points

Question 1 of 1

Use mathematical induction to show that for every positive integer n, 3 evenly divides $2^{2n} - 1$.

10.0 Points

Base case: when n=1, 2^(2n) - 1 = 2^2-1 = 4-1 = 3

Induction hypothesis: we assume 2^(2*(n-1))-1 is a multiplier of 3, as its case is proven when n is the smallest positive integer.

2^(2*(n-1))-1 = 4^(n-1) - 1 = 4^n/4 - 1

4^n - 1 = (4^n/4 - 1 + 1)*4 - 1. Since the first 2 items in the parentheses are a multiplier of 3, it leaves nothing but 1*4-1 =3, which is a multiplier of 3. A multiplier of 3 added by 3 is always a multiplier of 3.

As 4^n is nothing but 4* (4^n/4), 4^n must also be a multiplier of 4, leading to (4^n - 1)

Part 3 of 6 -

20.0 Points

Question 1 of 2

10.0 Points

A fair coin is tossed three times. Let X be the random variable that denotes the square of the number of heads. For example, in the outcome HTH, there are two heads and X = 4.

a. Find the distribution of *X*. That is, for each possible value of X, say what is the probability *X* would get that value.

b. What is E(X)? That is, find the expected value of X. Explain your answer.

a) There are 4 possible values of X, when there are 0, 1, 2, or 3 heads when tossed. Possible values of X are thus 0, 1, 4, or 9.

Probability of getting 0 head: C(3,0) * 0.5^0 * 0.5^3 = 0.125
Probability of getting 1 head: C(3,1) * 0.5^1 * 0.5^2 = 0.375
Probability of getting 2 heads: C(3,2) * 0.5^2 * 0.5^1 = 0.375
Probability of getting 3 heads: C(3,3) * 0.5^3 * 0.5^0 = 0.125

The distribution of X is thus the set of all possible combinations of value and its associated probability, namely, {(0, 0.125), (1, 0.375), (4, 0.375), (9, 0.125)}

b. The expected value is the probability-weighted sum of all possible values, namely:
0.125*0+0.375*1+0.375*4+0.375*9 = 5.25

Comment: wrong answer for part b, should be 0.125*9 or (1/8)*9 not 0.375*8.

Question 2 of 2

10.0 Points

Consider a blue spinner that has an outcome of 1, 2, 3, 4, or 5 and a red spinner that has an outcome of 1, 2, or 3. Each spinner is fair so that each outcome is equally likely.

Let X be the sum of the outcomes of the two spinners. What is E[X]? Explain your answer.

Hint: You may want use the linearity of expectations property

As each spinner is fair, the expected values of the outcome from the blue and red spinner are the arithmetic averages of the possible outcomes, namely, (1+2+3+4+5)/5 = 3 for blue spinner and (1+2+3)/3 = 2 for red spinner.

As both spinners are independent, and X is the sum of the expected values from them, E(X) is 3+2 = 5.

Part 4 of 6 -

15.0 Points

Question 1 of 1

You are given the following code:

15.0 Points

```
int main() {
    int n;
    int* arr;
    int arrSize;

    cout<<"Please enter a positive integer:"<<endl;
    cin>>n;

    arr = firstPosIntsArr(n, &arrSize);  // calling your function

    cout<<"The first "<<arrSize<<" positive integers are: ";
    printArray(arr, arrSize);
    cout<<endl;

    delete []arr;
    return 0;
}

void printArray(int arr[], int arrSize){
    int i;
    for(i = 0; i < arrSize; i++)
        cout<<arr[i]<<" ";
}
```

In this question, you should implement the function **`firstPosIntsArr`**, that given a positive integer **`n`**, it would create and return an array of the first **`n`** positive integers.

Define the function in a way that would allow it to work with the code above. That is, when executing the program, you could have the following interaction with the user:

```
Please enter a positive integer:
5
The first 5 positive integers are: 1  2  3  4  5
```

Hint: First, think how to define the prototype (header line) of the function, so it would fit to the way it is used in the program. Then, implement the body.

Notes:
1. You may use Xcode or Visual Studio to solve this question. You should create a new project, and work only in it. You are not allowed to look at old projects.
2. **For submission, download the file below. Paste your code in it, and upload it back.**

part 4.txt 0 KB

part 4.txt (0.8 KB)

Comment: Would not work for n > 10

Part 5 of 6 - 25.0 Points

Question 1 of 1 25.0 Points

Implement the function:

`int* findCommon(int arr1[], int arr2[], int n, int& outArrSize)`

This function gets two arrays of integers **`arr1`** and **`arr2`** with the same logical size **`n`**. We know that all elements in both arrays are taken from the range $\{1, 2, \ldots, n\}$.

When called, it should create and return a new array, that contains all the numbers that are both in **`arr1`** and in **`arr2`**. The function should also update the output parameter, **`outArrSize`**, with the logical size of the new array that was created.

Note: Each input array (**`arr1`** and **`arr2`**) could contain duplicate values. However, the output array **should not** have any duplicates. That is, a number that shows in both arrays (no matter how many time it shows in each one), should show in the output array only once.

For example, if **`arr1=[3, 1, 3, 2, 6, 4]`**, and **`arr2=[5, 4, 4, 3, 3, 3]`** after calling **`findCommon(arr1, arr2, 6, outArrSize)`**, the function should create and return an array that contains **`[3, 4]`**, and update the value in **`outArrSize`** to be **`2`**.

Implementation requirements: Your function should run in **linear time**. That is, it should run in $\theta(n)$.

Notes:
1. You don't need to submit a **main()** program.
2. You may use Xcode or Visual Studio to solve this question. You should create a new project, and work only in it. You are not allowed to look at old projects.
3. **For submission, download the file below. Paste your code in it, and upload it back.**

part 5.txt 0 KB

part5.txt (1.08 KB)

Part 6 of 6 - 15.0 Points

Question 1 of 1 15.0 Points

A sequence is called a **palindrome** if it reads the same backward and forward.. For example the sequence: 3, 12, 4, 4, 12, 3 is a palindrome. But, 3, 3, 7, 7 is not.

Give a recursive implementation for the following function:

`bool isPalindrome(int seq[], int seqSize)`

The function is given **`seq`**, an array containing a sequence of integers, and its logical size, **`seqSize`** . When called, it should return **`true`**, if **`seq`** represents a palindrome, or **`false`** otherwise.

Assumption: For simplicity, you may assume that the size of the sequence is even. That is, you may assume that **`seqSize`** is even.

Notes:

1. You don’t need to submit a `main()` program.

2. You may use Xcode or Visual Studio to solve this question. You should create a new project, and work only in it. You are not allowed to look at old projects.

3. For submission, download the file below. Paste your code in it, and upload it back.

part 6.txt 0 KB

part 6.txt (0.97 KB)

Comment: not a recursive implementation

Timezone: America/New_York

- Terms of Use
- Send feedback to the NYU Classes Team
- Powered by Sakai
-

Change Profile Picture

Error removing image

Error uploading image

Upload Choose File No file chosen

Save Cancel Connections ✖ Remove

Search for people ...

View More

My Connections Pending Connections

You don't have any connections yet. Search for people above to get started.

You have no pending connections.

←Back to My Connections

Search for people ...